Determining possible avenues of approach using ANTS

Pontus Svenson Hedvig Sidenbladh

Department of Data and Information Fusion
Division of Command and Control Systems
Swedish Defence Research Agency
SE 172 90 Stockholm, Sweden
ponsve,hedvig@foi.se
http://www.foi.se/fusion/

Abstract – *Threat assessment is an important part of level 3 data fusion. Here we study a subproblem of this, worst-case risk assessment. Inspired by agent-based models used for simulation of trail formation for urban planning, we use* ant colony optimization (ANTS) *to determine possible avenues of approach for the enemy, given a situation picture.*

One way of determining such avenues would be to calculate the "potential field" caused by placing sources at possible goals for the enemy. This requires postulating a functional form for the potential, and also takes long time. Here we instead seek a method for quickly obtaining an effective potential. ANTS, which has previously been used to obtain approximate solutions to various optimization problems, is well suited for this. The output of our method describes possible avenues of approach for the enemy, i.e, areas where we should be prepared for attack. (The algorithm can also be run "reversed" to instead get areas of opportunity for our forces to exploit.)

Using real geographical data, we found that our method gives a fast and reliable way of determining such avenues. Our method can be used in a computer-based command and control system to replace the first step of human intelligence analysis.

Keywords: ANTS, ant colony optimization, heuristical methods, swarm intelligence, determining avenues of approach, threat analysis, threat assessment, worst-case risk assessment

1 Introduction

Threat assessment is an important part of level 3 data fusion, as defined in [1]. The goal of level 2 data fusion [1], is to provide an accurate picture of current enemy activity. Given this information about the current time instant, the goal of threat assessment is to extrapolate it into the future to see which own objects are most threatened, and to determine which enemy objects pose the greatest threat. This is a complex task involving modeling the enemy's units, objectives, estimated knowledge about our forces, and doctrines. The role of threat assessment in level 3 data fusion and its relation to the worst-case risk assessment studied here is discussed further in section 2.1.

The algorithm presented in this paper is designed to give a worst-case scenario of what important locations the enemy objects can reach within certain time limits, given the terrain and the estimated mobility capabilities of the enemy objects. In section 3 we describe how the algorithm works.

Many important optimization problems can not be solved exactly in an efficient way (see, e.g., [2, 3]). For many problems there are, however, fast, approximate methods that will in most cases give a solution that is "good enough". One such algorithm, building on ideas from biology, is *ant colony optimization (ANTS)* [4, 5]. In nature, ants communicate by deploying pheromone (smell) paths that indicate the way between the ant-hill and food supplies. In the ANTS algorithm, ants are simulated agents who move through a search space or energy landscape looking for good locations. When an ant has found a sufficiently good place, it stops and distributes a pheromone along the path it took from its starting position. In our case, ants start from locations of enemy units as given by level 2 data fusion, and their goals are positions of own units. The ants will use the smell in addition to geographical and military-value information to determine where to move. The ANTS algorithm is described in more detail in sections 2.2 and 3. Since we assume that the enemy ants can determine when they have reached an own force-location, the output of our method will be a worst-case result.

Results of the ANTS method for a test scenario are shown in sections 4 and 5, where we give some suggestions for how to present the avenues of approach to a user. Section 6 compares the effective potential determined by ANTS with a calculated potential. Finally, sections 7 to 9 discusses our results and presents some ideas for future extensions of the method.

2 Background

2.1 Threat versus worst-case risk

Threat assessment is one of the most important and challenging parts of an information fusion system. The method presented in this paper gives a very fast way of obtaining possible avenues of approach given a situation assessment. It does not provide a complete threat analysis. Rather, the output of our algorithm is a map showing an "intelligent guess" as to what areas of the map the enemy may reach. The method will very quickly display a first guess and will then refine this incrementally until it converges to an approximation of the enemy's avenues of approach. Our algorithm basically simulates many imaginary enemy units doing local searches for own units; this is what makes it a worst-case risk assessment tool. An "exact" threat assessment would assume that the enemy has uncertain information, assign relevant probabilities or beliefs, and produce a ranked list of the enemy's probable objectives and the paths to them. Such a system could use our ant-based algorithm as a subsystem, combining the output of several runs of this subsystem using, e.g., some sort of Dempster-Shafer [6] or random set [7] formalism. Given a Dempster-Shafer belief function describing a situation picture, the ANTS method could be run for each of its focal elements. Attaching the corresponding probability mass to the resulting avenues would then produce a belief function over avenues of approach.

Despite the heuristical nature of ANTS, we argue that this kind of method is an important and indeed necessary part of an operational information fusion system. Our ant-based risk assessor provides a real-time indication of where enemy units might pose the biggest threat. Obtaining a threat analysis that is provably correct or correct with probability $1 - \epsilon$ using current methods requires a very large amount of computer resources. Our method, in contrast, runs in real-time and incrementally improves its output. Even in a command and control system that has a complete threat assessment module, an approximate solution such as ours has its place. The output of the fast, approximate algorithm can be used to aid in quick decisions. By comparing the output of this algorithm with the one from a slower but more accurate method, we can find out if the first suggestion was indeed correct. This is very similar to how humans often make a first guess, perhaps without knowing all the facts, and later refine their answer after thinking about the problem. It it also possible that a comparison between the worst-case analysis of ANTS and the action actually taken by the enemy could help us in determining how much the opposite side actually knows.

By having several different subsystems performing the same or nearly the same function, we also gain robustness for the overall system — some parts of it can break down and it will still be able to function. An additional benefit of having two or more subsystems performing the same task is that if the different methods give different results, something extraordinary (e.g., a completely new military strategy is adopted by the enemy) might be taking place and human operators/analysts need to look at the situation in more detail.

Our method can be used for worst-case risk assessment on all levels of force aggregation. It is just as easy to determine where a single tank might be headed as it is to determine where a battalion is going. Note that the geographical/locational information might actually be more accurate for a battalion since there are more restrictions where it can go (e.g., it needs a corridor of certain width).

2.2 Ants and Swarm Intelligence

ANTS is a form of collective, intelligent agent system. It is similar to other models for swarm intelligence and crowd behavior (e.g., [8, 9]). It was first used by Dorigo (e.g., [4]) to solve the traveling salesperson problem and has later been used for solving problems ranging from graph coloring [10] to routing and load balancing [11, 12]. The method shares a number of conceptual features with models for describing complex behavior such as Braitenberg vehicles [13] or Langton's vant [14].

ANTS is very similar to random walk or diffusion based methods for solving optimization problems, but adds interaction between the walkers to produce results more quickly. An ant is an agent that moves in the space of all solutions to the problem. The movement is basically a random walk but with one addition. An ant that reaches a good solution to the problem distributes a smell along the path it took to reach this solution. This smell is then sensed by other ants and increases their probability to move to a site with a high smell. This will lead to many ants being attracted to good sites. This process is similar to that used by real ants to communicate paths to food-sources; hence the name "ants" for the agents. ANTS is suited primarily for problems that have a natural representation in a low-dimensional metric space or a graph of not too high connectivity.

Note that the way we use ANTS here is different from traditional optimization. We do not solve an optimization problem. The smell, which gives the ants their ability to solve optimization problems, is here the output of the method. This smell determines avenues of approach, that is, locations where the enemy may move.

It is instructive to compare the ANTS method with tracking methods. The model for how the agents move is similar to that used in for example particle filtering (e.g., [15]), but adds some extra non-linearity in the form of the use of the smell. This leads to a different, more explicit, kind of interaction between the agents than in tracking models. In a way, we can say that the smell takes the place of observations when we attempt to predict future positions.

Active walker models similar to the ants used here have

previously been used to model formation of trails and crowd behavior (e.g., [8]). One application of this which is somewhat similar to our risk analysis is in urban planning. Another possible application of methods such as these is to crowd-control; here ANTS would be used to predict where a crowd will move.

3 The ANTS algorithm

Our method is best explained by examining a situation with one enemy unit. Start by inserting N ants at its location. In each time-step, each ant randomly selects a neighbor of its current location and moves there. The probability is not uniform over all neighbors. Instead, the type of the unit represented by the ant and the terrain is taken into account so that, e.g., it is more probable for a tank to move along a road than into a forest. It is also more probable to move into a position that has a high military significance, e.g., with own units nearby, on top of a hill, etc. The probabilities are also modified by smell/pheromone traces left there by other ants. Initially, there is no smell anywhere in the map. As soon as an ant reaches a favorable military position (e.g., one of the target units), it stops and distributes a smell along the way it took to reach its goal.

An ant can represent any of several different types of enemy units, from an infantry-squad to a battalion of tanks. The ants basically perform interacting random walks with probabilities that are site-dependent.

An ant at position $\mathbf{x}$ uses three kinds of information to determine its future position. First, we have geographical information $T(\mathbf{y},\mathbf{x})$ that simply says how long it would take the ant to reach each of the neighboring sites of $\mathbf{x}$. This information is predetermined and comes from a terrain database of the battlefield. It is different for ants that represent different kinds of enemy units. This information is taken as fixed in our current simulations, but it is straightforward to change this in real-time in order to implement changes in accessibility due to war activity (e.g., a bombed bridge should be reflected in this information).

The second component, denoted $F(\mathbf{y})$, is related to the strategic importance of different locations $\mathbf{y}$. This is highest where the targets that the ants try to reach (i.e., our units) are. A human operator could change this field using their intuition and experience of where the important part of the battle will take place.

The third part is the smell distributed by other ants, $S(\mathbf{y},t)$. At $t = 0$, this is initialized to 0 for all $\mathbf{y}$:

$$S(\mathbf{y},0) = 0. \tag{1}$$

The smell S will be updated during the run of the algorithm, see equation 4 below.

The probability to go to a site $\mathbf{y}$ from $\mathbf{x}$ at time t is thus given by

$$p(\mathbf{y},\mathbf{x},t) = 0 \tag{2}$$

1. while maximum time not reached
   (a) for all ants i
      i. set $\mathbf{x}$=current position of ant i
      ii. randomly select a neighbor of $\mathbf{x}$ using equation 3 and move ant i there
      iii. if ant i at target then
         A. update smell for all sites visited by ant i according to equation 4
         B. kill ant i
   (b) if $S(\mathbf{x})$ has not changed, exit loop
2. output smell as effective potential

Figure 1: Pseudo-code for the ANTS algorithm.

if $\mathbf{x}$ and $\mathbf{y}$ are not nearest neighbors, and

$$p(\mathbf{y},\mathbf{x},t) \propto \frac{1}{T(\mathbf{y},\mathbf{x})} + \omega_s S(\mathbf{y},t) + \omega_f F(\mathbf{y}) \tag{3}$$

otherwise. In equation 3, $T(\mathbf{y},\mathbf{x})$ is the geographical information regarding the time needed for the ant to move from $\mathbf{x}$ to $\mathbf{y}$, S and F are the smell and value fields introduced above, and $\omega_s = 1$ and $\omega_f = 1$ are weights determining the relative importances of the different fields. The constant of proportionality is determined by requiring that summing over all $\mathbf{y}$ gives unit probability.

A possible addition to the F field is to include also information on visibility and range of fire at different points, as is done in [16]. The drawback of doing this is that it adds to the storage and/or computation requirements for the background fields, thus destroying the attractive simplicity of the ants model.

In order to avoid loops, we added a small bias against moving back to where the ant came from. If the new position is equal to the ant's old position, a new random number is drawn and a new position is determined from it. This reduces the probability for the ant to go back in its own track quadratically. We choose not to completely disallow backmoves in order to avoid an ant getting trapped at a location with only one viable exit. The random number generator used in all simulations was Matlab's **rand**-function.

All ants are time-evolved in parallel. As soon as an ant reaches a target site it will stop and a trace of smell will be distributed along the way it took to reach this target. If the path that the ant has traveled is given by the sequence $\mathbf{x}_i$ for $i = 0,\ldots,M$ (with $\mathbf{x}_0$ equal to the starting position), we change the smell field according to

$$S(\mathbf{x}_j,t+1) = S(\mathbf{x}_j,t) + \frac{j}{M}. \tag{4}$$

For sites $\mathbf{x}$ not visited by the ant, the old value for S is propagated to time $t+1$. This smell will be used by other ants to determine their future time-evolution (and also displayed

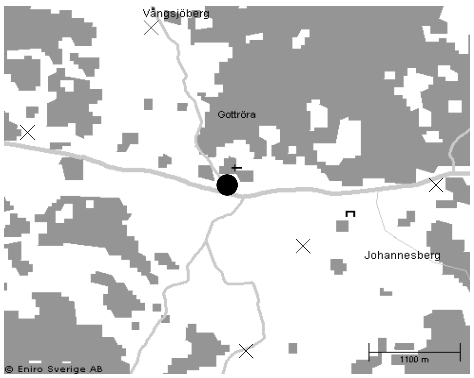

Figure 2: The example map used for our tests. The black circle marks the location of the enemy unit whose movement we are trying to predict, black x:es show locations of own units. The different colors of the terrain indicate different mobilities for the enemy unit.

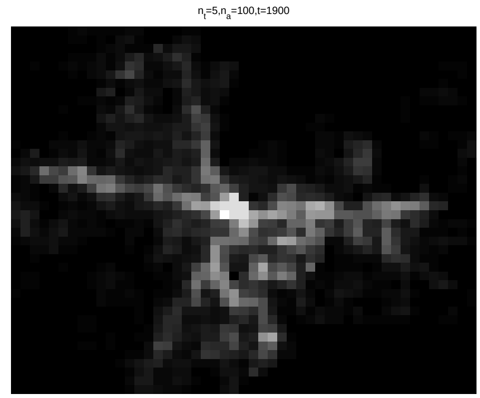

Figure 3: The distribution of smell after 1900 time-steps for the scenario with 5 targets and 100 ants using a linear gray-scale.

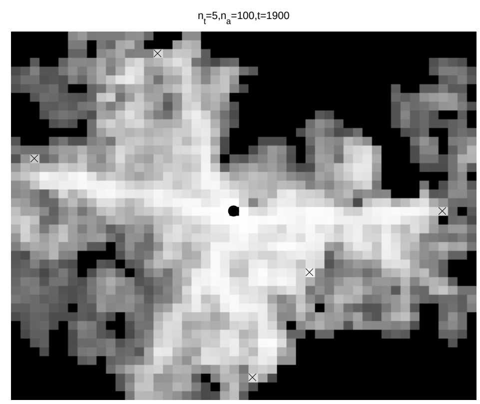

Figure 4: The distribution of smell using a gray-scale determined by histogram equalization after 1900 time-steps for the scenario with 5 targets and 100 ants.

to the user as an indication of where the interesting areas in the map are). A number of possible extensions can be made here: the ant could continue from the target when it has distributed its smell or it could be restarted at the start position.

Pseudo-code for the algorithm is shown in figure 1.

The output of the program is not the final positions of the ants but rather the effective potential $S(\mathbf{x})$ determined by the distribution of smell on the map. This distribution will of course change as more and more ants reach the targets. This is an important feature of the algorithm: it will run in real-time and provide incrementally better and better approximations to the threat analysis.

4 Results

The test scenario presented here takes place on the map shown in figure 2, where we also show the start position (black circle, close to the center of the map) and the positions of five targets (black x:es, one at the middle of each side of the square and one below and to the right of the center). The enemy unit at the black circle is assumed to have different mobilities on the road (light grey), in the field (white) and forests (dark grey). In the simulations presented here, the mobility in the forests is considerably smaller than that in the other types of terrain, leading to almost zero probability of entering such areas.

In figure 3, we present the "smell" left by the ants that is the output of the method. Smells of different strengths are presented using different gray-scales, where white represents the largest amount of smell in the figure and black the lowest. For visibility we chose to map the smell onto a non-linear gray-scale using a histogram equalization method (see, e.g., [17]) to determine the appropriate mapping from smell to gray-scale. The results of this transformation for the distribution shown in figure 3 is shown in figure 4. (In practice, the transformation works like a logarithmic gray-scale — more smell-values are mapped to the same grayness at large smells than at lower. It works almost like a filter that filters out all smells higher than some threshold. We found that this gave the best representation of the distribution for this medium.)

By comparing figure 4, which shows the smell after convergence, to figures 5 to 7 which show the smell at earlier times, we can see that the convergence is rather quick for most areas of the map. Note however the anomalous behavior near the top target. At times 100 and 200, the distribution of smell indicates that the enemy would take a detour going first left along the road and then up through fields to reach the top. At time 700, the method has discovered the "correct" avenue of approach to this target. This is a clear indication of the heuristic nature of the algorithm showing

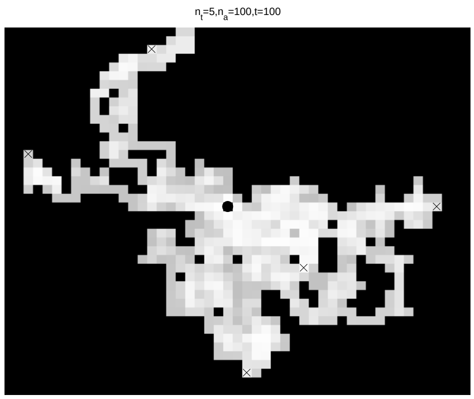

Figure 5: Distribution of smell using a gray-scale determined by histogram equalization after 100 time-steps for the scenario with 5 targets and 100 ants.

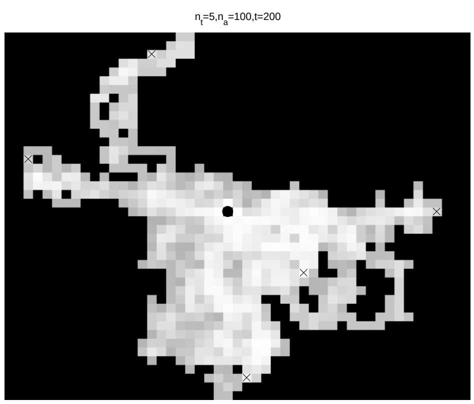

Figure 6: Distribution of smell using a gray-scale determined by histogram equalization after 200 time-steps for the scenario with 5 targets and 100 ants.

both the need to run ANTS until convergence and how it incrementally improves its output.

We have also run tests using different numbers of ants. In figure 8, we show the same scenario but using just 30 ants. It is clear that this is too small a number of ants to be able to provide an accurate risk analysis: the upper target can not be found since the ants get stuck near the center.

Figure 9 shows that using too many ants, in this case 10^3, also does not lead to a good convergence of the smell. In this case it is probably due to too much smell being released near the start, which causes many of the ants to get trapped here. After some experimentation, we tentatively recommend using on the order of 100 ants for each simulation. If there are many targets, it is better to divide the targets into sets containing on the order of 5 targets each and run a separate simulation for each set. The results can then be combined for display in the command and control system.

We have also made tests varying the number and loca-

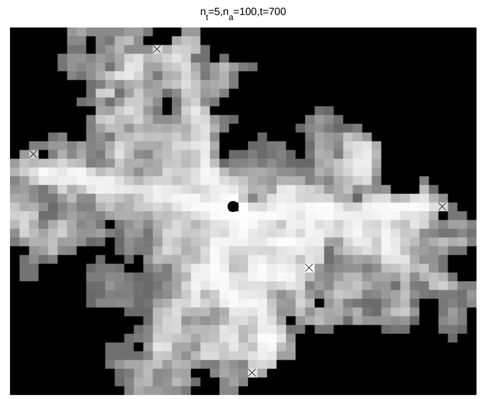

Figure 7: Distribution of smell using histogram equalization after 700 time-steps for the scenario with 5 targets and 100 ants.

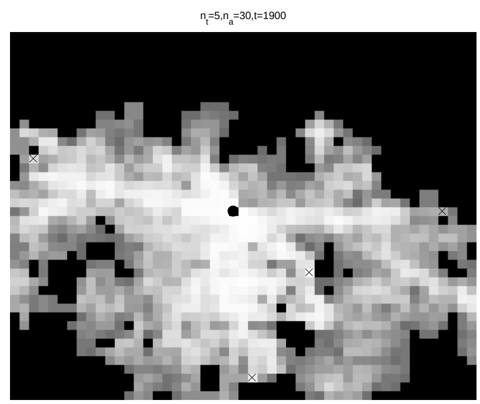

Figure 8: Distribution of smell using a gray-scale determined by histogram equalization after 1900 time-steps for the scenario with 5 targets and 30 ants.

tions of the targets and found similar results to those in the scene presented here. As the number of targets and start positions increases, the probability distribution for future locations will be fuzzier and fuzzier. This also means that it will we harder and harder for a human analyst to determine what is happening. The ANTS algorithm provides hints for where the human should concentrate their attention. For analyzing a complex scene with perhaps dozens or hundreds of interesting goals, it is best to use only a few targets (2-5) at a time and instead run several simulations and combine their output. In this way, the ANTS method will be able to provide a rough guide to where the human analyst should focus their attention.

None of the simulations presented here took more than a minute to run using Matlab 6.5 on an 1.4GHz AMD Athlon CPU, and most of the CPU-time was spent in Matlab's functions for displaying graphics .

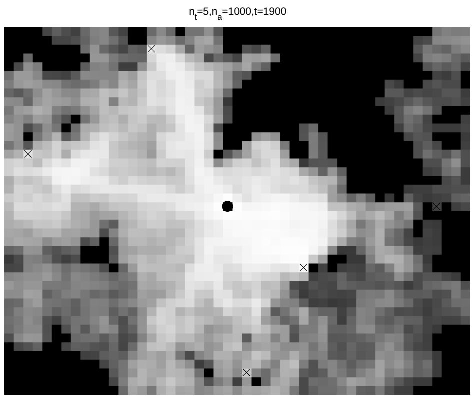

Figure 9: Distribution of smell using a gray-scale determined by histogram equalization after 1900 time-steps for the scenario with 5 targets and 1000 ants.

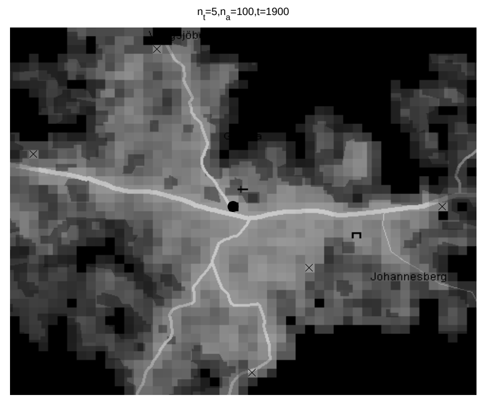

Figure 10: One way of combining histogram-equalized smell and map reminiscent of representations used in computer games. Data is shown for the scenario with 5 targets and 100 ants after 1900 time-steps.

5 Visualizing possible avenues of approach

In order to show avenues of approach on the map, we converted the map to a gray-scale representation, using darker shades of gray for areas with smaller mobilities. All subsequent figures show what this map looks like using the smell as a filter to change the gray-scale. We used two different ways of combining the map and the smell. In the first (shown in figure 10), the grayness of a pixel is the product of the map's grayness at that location and the grayness determined by the smell. In practice, this means that areas of the map where no ant has left any smell will be blacked-out, while those areas that have the most smell (i.e, that the ANTS algorithm consider most interesting) will appear normal. The purpose of this is to draw the user's attention to the avenues of approach, while the black portions require less attention. It is instructive to compare this with the way the map in computer strategy games like *Civilization* starts out black and then becomes visible only after the area has been explored by the player.

The second way of combination (figure 11) simply displays the maximum of the map-value and the smell-value at each pixel. This is a better representation since it shows more clearly where the ants move and is less influenced by the roads. The combination using multiplication is better for showing what areas the enemy can reach, while the max-combination better shows the relative differences in occupation probability, and can hence be used for determining where to increase surveillance or attack.

Note that the particular way we present the results is of course not a part of the algorithm. A real implementation would use colors and shading to combine these representations and also allow the user to display, e.g., only those places where the smell is larger than some cut-off value.

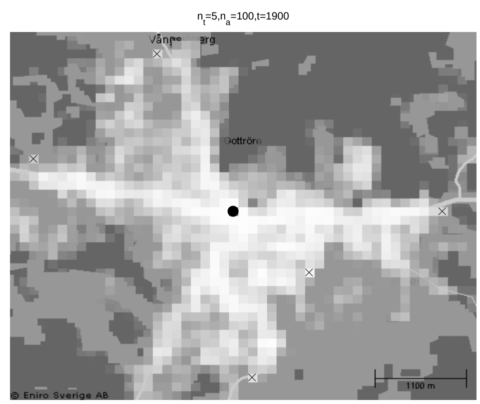

Figure 11: This figure shows max of histogram-equalized smell and map after 1900 time-steps for the scenario with 5 targets and 100 ants.

6 An "exact" potential instead of ANTS

An alternative to using ANTS to determine the effective potential induced by the targets on the terrain is to place sources at the target locations and calculate the exact potential at all locations in the map, taking into account also the terrain. Assuming Gaußian sources of strengths K_n and with widths σ_n, the potential at location $\mathbf{x}$ would then be

$$U_0(\mathbf{x}) = \{\sum_n K_n \exp(\frac{-\parallel \mathbf{r}_n - \mathbf{x} \parallel^2}{2\sigma_n})\}(1 - \frac{\parallel \mathbf{x} - \mathbf{r}_0 \parallel^2}{d^2}), \tag{5}$$

where $\mathbf{r}_n$ are the goal positions, $\mathbf{r}_0$ the start position and d the diameter of the map. Note that this potential as well as all other fields used in this paper is assumed to live on the set of points $\mathbf{x}$ of a discretized lattice representation of

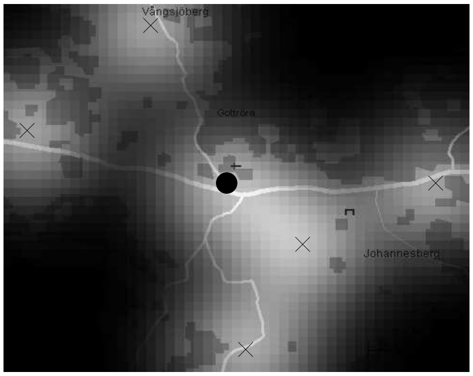

Figure 12: Terrain map modified by potential from targets.

the area of operations. Note that the exact expression for the potential in equation 5 is of course completely arbitrary — we choose this modified Gaußian for simplicity. It has nothing to do with the ANTS methods presented here.

Calculating any such potential takes time

$$T_d \sim N_t L^2, \tag{6}$$

where N_t is the number of targets and L the linear size of the map. The time needed for the ANTS algorithm to determine the effective potential is

$$T_a \sim N_a T_c, \tag{7}$$

where N_a is the number of ants used and T_c is the time needed to reach convergence. For diffusion-based methods without interaction, T_c would scale as

$$T_c \sim \langle \| \mathbf{r}_t - \mathbf{r}_s \|^2 \rangle_{\mathbf{r}_s}, \tag{8}$$

where $\mathbf{r}_s$ is the starting position and the average is over all target positions $\mathbf{r}_t$. The presence of the smell in equation 7, however, leads to a behavior more like that of super-diffusive dynamics,

$$T_c \sim \max_{\mathbf{r}_s} \{ \| \mathbf{r}_t - \mathbf{r}_s \|^\alpha \}, \tag{9}$$

with α close to 1. Since

$$\max_{\mathbf{r}_s} \{ \| \mathbf{r}_t - \mathbf{r}_s \|^\alpha \} \sim L^\alpha, \tag{10}$$

it is clear that ANTS gives a large speed-up over calculating the exact potential.

Figure 12 shows the terrain map modified by sources at the target locations, equation 5, using sources of equal strength 4 and with standard deviation 30. Comparing figure 12 to figure 3, it is clear that the effective potential determined by the ANTS method is a good approximation to this U_0. To get a measure of the speed of convergence of ANTS, we calculated the discrete Kullback-Leibler [18] distance

$$K[f, g; t] = \sum_{\mathbf{x}} g(\mathbf{x}, t) \log \frac{g(\mathbf{x}, t)}{f(\mathbf{x})}, \tag{11}$$

using normalized distributions

$$f(\mathbf{x}) = \frac{U_0(\mathbf{x})}{\sum_{\mathbf{y}} U_0(\mathbf{y})} \tag{12}$$

and

$$g(\mathbf{x}, t) = \frac{S(\mathbf{x}, t)}{\sum_{\mathbf{y}} S(\mathbf{y}, t)}. \tag{13}$$

We found that the Kullback distance decreased exponentially with time, stabilizing after about 500 time-steps at a value about an order of magnitude smaller than at $t = 0$. Information such as that displayed in figure 12 could be used to help humans focus on the most important areas of the map. The exact appearance of equation 5 is of course completely arbitrary. In addition to being much faster, the ANTS method also does not require us to postulate any such expression for the potential: it only requires the locations of targets and the parameters governing the distribution of smell in equation 4.

7 Discussion

Conventional threat analysis takes the current situation and uses our knowledge of where our important assets are to try to predict where the enemy is headed. ANTS, in contrast, flips the sides: we try to predict what the enemy will do by putting ourselves in their position and determining what we would do, without assuming that the enemy has global information about our assets and positions. We argue that using a local-search method in this way gives a more robust threat prediction, since the output is determined by simulating the enemy, not by trying to guess their objectives. The ANTS algorithm can be easily adapted to new information regarding enemy behavior (by changing equation 3). This is important, since potential enemies will also have computer systems to aid them, probably leading to more surprising tactics.

Another goal of using ANTS is to minimize the amount of work needed by humans. Given a terrain map and the locations of enemy forces, humans can often determine possible avenues of approach visually, and then decide where to concentrate own forces and sensors in defense. Our method does not aim to completely replace such human analysis, but can act as a first step by suggesting such avenues to the analyst. The output of our program, together with the output from the situation assessment routines that are used as inputs to our program, help the human operators to focus only on the most important parts of the map. In addition, since the ANTS algorithm is meant to run interactively and provide incrementally better distributions as time goes, it can also be adapted to give an answer to the question of what would happen if the enemy suddenly receives some new information.

8 Future extensions

Our current ANTS method can be extended in a number of ways. It is, for instance, possible to add some movement of the targets (i.e., our units) or to include changes in mobility caused by blowing up bridges.

In the simulations presented here, we have used just one type of object to track at all times. It is straightforward to extend the method so that it can handle situations where it is given several different types of objects (e.g., a platoon of tanks at position $\mathbf{x}$ and a company of infantry at position $\mathbf{y}$) as input to get the combined threat posed by all of these. Ants with different mobilities should then be started at each of the enemy positions, and the output should be changed so that it gives smells for all types of objects. Ants should here be attracted primarily to smell of its own type, but also to that of other types. This makes it possible to model things like tanks following scout patrols of infantry, or infantry following tanks.

The ANTS method as presented here can also be used for the more interesting problem of "opportunity analysis", i.e., to determine what possibilities own forces have given an accurate situation picture. We are planning to study these and other extensions to ANTS in future work.

9 Conclusions

In conclusion, we showed how ANTS can be used to get a quick worst-case risk assessment. By simulating the enemy instead of relying on static assumptions of their objectives, we obtain a method that is more robust if the enemy also uses computer-assisted command and control systems. The ANTS method should be integrated in a command and control system and provide a first, real-time indication of avenues of approach. More thorough analysis methods should also be part of this system and will give an updated more exact picture at some later time. The system should also contain modules that automatically compare the quick picture with the reliable one, and warns the human operators when they differ by too much.

The possible avenues of approach that is the output of the ANTS method can also aid in sensor allocation and management, to help determine which areas should be surveyed by sensors such as UAV's.

Acknowledgements: We thank Christian Mårtenson and Johan Schubert for helpful discussions.

References

[1] D. L. Hall and J. Llinas, editors. *Handbook of Multisensor Data Fusion*. CRC Press, Boca Raton, FL, USA, 2001.

[2] Christos H. Papadimitriou. *Computational Complexity*. Addison-Wesley, Reading, MA, 1994.

[3] Michael R. Garey and Davis S. Johnson. *Computers and Intractability. A guide to the Theory of NP-Completeness*. W H Freeman, New York, 1979.

[4] M Dorigo, V. Maniezzo, and A. Colorni. The Ant System: Optimization by a Colony of Cooperating Agents. *IEEE Transactions on Systems, Man, and Cybernetics-Part B*, 26(1):29–41, 1996.

[5] E. Bonabeau, M. Dorigo, and G. Theraulaz. *Swarm Intelligence: From Natural to Artificial Systems*. Oxford University Press, Oxford, UK, 1999.

[6] Glenn Shafer. *A mathematical theory of evidence*. Princeton University Press, 1976.

[7] I.R. Goodman, Ronald P.S. Mahler, Hung T. Nguyen. *Mathematics of Data Fusion.* Kluwer Academic Publishers, 1997

[8] Dirk Helbing, Frank Schweitzer, Joachim Keltsch, and Peter Molnar. Active walker model for the formation of human and animal trail systems. *Physical Review*, E56:2527–2539, 1997.

[9] E Bonabeau and G. Théraulaz. Swarm smarts. *Scientific American*, pages 72–79, March 2000.

[10] D. Costa and A. Hertz. Ants Can Colour Graphs. *J. Oper. Res. Soc.*, 48:295–305, 1997.

[11] R. Schoonderwoerd, O. Holland, J. Bruten, and L. Rothkrantz. Ants for load balancing in telecommunication networks. Technical Report HPL–96–35, Hewlett-Packard Laboratories, Bristol, UK, 1996.

[12] I. Kassabalidis, M. A. El-Sharkawi, R. J. Marks II, P. Arabshahi, and A. A. Gray. Swarm intelligence for routing in communication networks. In *IEEE Globecom*, 2001.

[13] Valentino Braitenberg. *Vehicles: Experiments in Synthetic Psychology*. MIT Press, 1986. reprint edition.

[14] C. G. Langton. Studying artificial life with cellular automata. *Physica*, D22:120, 1986.

[15] N. Gordon, D. Salmond, and A. Smith. A novel approach to nonlinear/non-Gaussian Bayesian state estimation. *IEE Proceedings on Radar, Sonar and Navigation*, 140(2):107–113, 1993.

[16] R. Richbourg and W. K. Olson. A hybrid expert system that combines technologies to address the problem of military terrain analysis. *Expert Systems with Applications*, 11(2):207–225, 1996.

[17] Rafael C Gonzales and Richard E Woods. *Digital Image Processing*. Addison-Wesley, 1993.

[18] Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory.* Wiley, 1991.